APPLICATION

FOR

UNITED STATES PATENT

Entitled

A SYSTEM AND METHOD TO EXCHANGE INFORMATION BETWEEN A CONTROL ELEMENT AND FORWARDING ELEMENTS IN A NETWORK ELEMENT ARCHITECTURE

Inventor:

Hormuzd M. Khosravi

David W. Rouille
Daly, Crowley & Mofford, LLP
275 Turnpike Street, Suite 101
Canton, Massachusetts 02021-2310
Telephone (781) 401-9988 x24
Facsimile (781) 401-9966

Intel Corporation
Intel Case No.: P18639
Attorney Docket No.: INTEL-022PUS

TITLE

A System and Method to Exchange Information Between A Control Element And Forwarding Elements In A Network Element Architecture

CROSS REFERENCE TO RELATED APPLICATIONS

Not Applicable

STATEMENT REGARDING FEDERALLY SPONSORED RESEARCH

Not Applicable.

FIELD OF THE INVENTION

The present application relates generally to a network element architecture and, more particularly, to a mechanism for exchanging information between a control element and forwarding elements in the network element architecture.

BACKGROUND OF THE INVENTION

In recent years, a trend has emerged in the networking industry in which Internet Protocol (IP) network elements (NEs) have begun to evolve from highly customized and integrated designs to modular pieces of equipment. NEs appear to external entities as a monolithic piece of network equipment, such as a router, network address translator (NAT), firewall, or load balancer. Internally, however, the NE (such as a router) is composed of numerous logically separated entities that cooperate to provide a given functionality (such as routing).

Two types of network element components are commonly used in a NE: a control element (CE) and multiple forwarding elements (FEs). Control elements typically provide control functionality e.g., routing and signaling protocols. Forwarding elements typically handle data path operations for each packet.

There are several known mechanisms used for exchanging information between control elements and forwarding elements in NEs. Most of these mechanisms are FE and/or interconnect specific. Typically, the exchange of information between an FE and a CE occurs across a single channel carrying both control and configuration messages as well as data packets. Further, the messages tend to be large in size. The NE may also be susceptible to Denial of Service (DoS) attacks in which the network element is flooded with bogus control protocol packets, which saturate the channel established between the FE and the CE.

BRIEF DESCRIPTION OF THE DRAWINGS

The present mechanism for exchanging information between a control element and a set of forwarding elements will be more fully understood from the following detailed

description taken in conjunction with the accompanying drawings, in which:

Figure 1 comprises a block diagram of a network;

Figure 2 comprises a block diagram of a network element architecture;

Figure 3 comprises a block diagram of a control element and a forwarding element;

Figure 4 comprises a block diagram of a router architecture; and

Figure 5 comprises flow chart showing the implementation of a network element protocol.

DETAILED DESCRIPTION OF THE INVENTION

Before describing the present mechanism for exchanging information between a control element and a set of forwarding elements, some introductory concepts and terminology are explained.

A Forwarding Element, Control Element Separation (ForCES) protocol "ForCES Protocol" refers to the ForCES post-association phase protocol. The ForCES Post-Association Phase Protocol is a protocol used for post-association phase

communication between CEs and FEs (defined below). The ForCES Protocol is a master-slave protocol in which the CEs are masters and the FEs are slaves. This protocol includes both the management of the communication channel (e.g., connection establishment, heartbeats) and the control messages themselves. This protocol could be a single protocol or could include multiple protocols working together, and may be unicast-based or multicast-based.

A forwarding element (FE), sometimes referred to as a forwarding plane, comprises a logical entity providing per-packet processing and handling as directed by a CE. Functions performed by the FE include LPM (longest prefix matching) packet forwarding, packet classifying, traffic shaping, metering, and network address translating. In one embodiment the FE implements the ForCES protocol.

A control element (CE), sometimes referred to as a control plane, comprises a logical entity instructing one or more FEs regarding how to process packets. CEs handle functionality such as the execution of control and signaling protocols. Some examples of control functions that can be implemented in the CE include routing protocols as well as control and signaling protocols. In one embodiment the CE implements the ForCES protocol.

The Network Processing Forum (NPF) Application Programming Interface (API), published August 2002, is a programming interface to control plane applications. The NPF API makes the existence of multiple forwarding planes as well as vendor-specific details transparent to the control plane applications. Thus, the protocol stacks and network processors available from different vendors can be easily integrated with the NPF APIs.

The Routing Information Protocol (RIP) comprises an interior gateway protocol that specifies how routers exchange routing table information. The Open Shortest Path First (OSPF) protocol is an interior gateway routing protocol developed for IP networks based on the shortest path first or link-state algorithm. Routers use link-state algorithms

to send routing information to nodes in a network by calculating the shortest path to each node based on a topography of the network constructed by each node. Each router sends the portion of the routing table that describes the state of the router's own links.

Type-Length-Value (TLV) is a method of encoding parameters within a protocol message. TLV encoding typically includes one octet of type (T), one octet of length (L) and octets of value (V). The Type field indicates the type of items in the Value field. The Length field indicates the length of the Value field. The Value field comprises the payload portion of the packet.

Generic Routing Encapsulation (GRE) is an encapsulation protocol used for encapsulating packets inside a network layer protocol such as IP.

Referring now to Figure 1, a block diagram of a network is shown. The network includes a first network router 1 coupled across a local area network (LAN) 20 to three systems 3, 4, and 5. The systems 3, 4, and 5 may be work stations, desktop computers , laptop computers or the like. The first router 1 is also coupled to a second router 2 which is coupled to three systems 6, 7, and 8 across LAN 22. First router 1 includes a control element 101 and multiple forwarding elements 102, 103, and 104. Second router includes a control element 105 and multiple forwarding elements 106, 107 and 108. With such an arrangement packets can be transferred from one system to another. For example, system 4 can transfer packets to system 8 by forwarding the packet to forwarding element 102 resident in first router 1. The control element 101 of first router 1 directs the packet received in forwarding element 102 to forwarding element 104. The packet is directed out of router 1 from forwarding element 104 to forwarding element 106 of second router 2. The control element 105 of second router 2 directs the packet from forwarding element 106 to forwarding element 108. The packet is then directed from forwarding element 108 to system 8.

Referring now to Figure 2, network element 10, such as a switch or a router, typically include a control element 100 and multiple forwarding elements 110a and 110b. The control element 100 controls and configures the forwarding elements 110a and 110b and the forwarding elements110a and 110b manipulate the network traffic. In general, the control element 100 executes signaling and/or routing protocols (e.g. RIP, OSPF) and provides control information to the forwarding elements 110a and 110b. The forwarding elements 110a and 110b make decisions based on this information and perform operations on packets such as forwarding, classification, filtering, and the like.

The standardized Application Program Interfaces (APIs) within the control element 100 and forwarding elements 100a, 100b enable system vendors, Original Equipment Manufacturers (OEMs), and end users of network elements to mix and match components available from different vendors to achieve a device of their choice. The Network Processing Forum (NPF) API is an industry standard API for this purpose and provides a flexible and well-known programming interface to the control plane applications. The NPF API makes the existence of multiple forwarding elements, as well as vendor-specific details, transparent to the control element applications. Thus, the protocol stacks and network processors available from different vendors can be readily integrated with the NPF APIs.

In the illustrative embodiment, a NE includes a CE 100, a Transport Plugin 30 and FEs 110a and 110b. The CE 100 is used to provide the implementation for the NPF APIs described above. The CE 100 is aware of multiple FEs 110a and 110b and has the intelligence to perform one-to-many mapping of the FEs for the application. The transport plugin 30 facilitates communication between the CE and FEs and implements the transport protocol.

The presently described protocol provides a mechanism for exchanging information between the CE 100 and one or more FEs 100a, 110b in a network element

architecture. In a particular embodiment the protocol is referred to as the FLEX (ForCES Light-weight Extensible) protocol. The FLEX protocol may form a part of the Transport Plugin 30.

The inventive protocol is a stateless, request-response protocol between the control element 100 and the forwarding elements 100a, 110b in an NE. The protocol is relatively lightweight in terms of both low message parsing overhead and utilization of small message sizes, which may be accomplished by using TLV or compact binary encapsulation for the message or packet payload. In one embodiment, the protocol has a fixed length header that is 8-bytes long followed by a variable length payload, with the messages being 32-bit aligned. An example packet header is shown below:

Version	Flags	Message Type	Command Correlator

The version field is one byte long and contains the version number of the protocol being used. The flags field is also one byte long and contains various flags that are used in protocol headers. The message type field is two bytes long and defines the message type (e.g. bind request, bind response, etc.). The command correlator field is four bytes long and is used to distinguish between commands of the same type. The command correlator field contains a sequence number for the command which is used in order to distinguish between the response to the commands of the same type. For example, a CE may issue a bind request to a first FE and a bind request to a second FE. The command correlator field assigns a different sequence number to each bind request such that the response to the first bind command can be distinguished from the response to the second bind request.

The inventive protocol is extensible by separating the data model from the base protocol. Thus, the message functionality is defined by the inventive protocol while the data within the message is defined by the data model. The inventive protocol provides

the base functionality messages (e.g. configure request) while data that is included inside the messages (e.g. an IP forwarding table) is defined by the data model. Since the inventive protocol is separate from the data model, the inventive protocol is extensible and can carry different types of information. A separate FE model defines the data that needs to be exchanged or the payload for the inventive/base protocol.

The protocol implements separate control and data channels. Referring again to Figure 2, the data channel 140, 144 carries control protocol packets, such as RIP or OSPF packets, which are either redirected from the FEs 110a, 110b to the CE 100 or are to be forwarded from the CE 100 to another FE. The control channel 142, 146 carries other control and configuration messages. The present protocol's separation of the control channel from the data channel allows the use of different transports for the control channel than those transports used in the data channel. The present protocol supports different interconnects by allowing different encapsulations for different interconnects. For example, when an IP interconnection is used, the inventive protocol uses TCP for the control channel. The present protocol meets ForCES requirements including command bundling, message priority, dynamic association and failover support.

The separation of the control and data channel helps make the present protocol robust against Denial of Service (DoS) attacks. In a DoS attack, a malicious user tries to bring down the system or network element by flooding the network element with bogus control protocol packets. If a single channel is used to carry both control messages from CE to FE as well as control protocol packets from FE to CE, the single channel would be overwhelmed by the bogus control protocol packets in case of a DoS attack. By separating information transferred between the CE 100 and the FEs 110a, 110b into two separate channels, the control channel can continue to work properly even when the data channel is overwhelmed during a DoS attack. Thus, the protocol is more robust against DoS attacks.

In one embodiment, the information exchange between the CE 100 and FE 100a, 110b using the present protocol includes multiple phases: a binding phase, a capability and topology discovery phase (also referred to as a capability discovery phase), and a configuration operation phase.

Figure 3 shows an exemplary information exchange for these phases. In the binding phase, the FE 110 sends a bind request 112 to the CE 100. In response to receiving the bind request 112, the CE 100 sends back a bind response 114 to the FE. The bind response indicates whether the bind was successful or not. During this phase, encapsulation information is exchanged between the CE 100 and the FE 110, which leads to the provision of a separate data channel, such as a GRE tunnel, for the exchange of data packets between the CE 100 and the FE 110.

In the capability discovery phase, the CE 100 sends a capability request 116 to the FE 110. The FE 110, in response to receiving the capability request 116, sends back a capability response 118 with its capability information to the CE 100. For example, information such as the FE can perform IPv4 forwarding, 5-tuple packet classification, etc. The CE 100 also sends a topology request 120 to the FE 110. The FE 110 responds to the topology request 120 with a topology response 122 which includes the FE's topology information relative to other FEs. For example, information such as FE A is directly connected to FE B, C. If the CE 100 deems the FE's capabilities and topology acceptable and if the CE 100 is ready to control and configure the FE 110, the CE 100 sends a start FE operation message 124 to the FE 110. After the start FE operation message is received by the FE 110, the FE 110 can report events and send packets to the CE 100. A heartbeat message exchange also starts after the start FE operation message 124 is sent. The heartbeat is a periodic "pinging" of the CE and FE to ensure that they are still active. If the CE 100 is not capable of controlling or configuring the FE based on the FE's capabilities or topology, the CE 100 sends an unbind message 138 to the FE at this point.

In the configuration operation phase, a configuration request 126 is sent from the CE 100 to the FE 110. In response to receiving the configuration request 126, the FE 110 sends back a configuration response 128 to the CE 100. The configuration response 128 will indicate whether the configuration request 126 was successful or it failed for some reason. The CE 100 also sends a query request 130 to the FE 110. In response to receiving the query request 130, the FE 110 sends back a query response 132 to the CE 100. For example, the CE could send a query request 130 to query the status of a port on the FE 110 and the query response 132 would indicate the status of that port. Asynchronous FE events 134, such as port down events, are also reported to the CE 100. Packet redirection 136 between the CE 100 and the FE 110 also takes place i.e. control packets such as RIP, OSPF messages are redirected to the CE 100 from the FE 110 and vice-versa over the Data channel. Heartbeat messages are also exchanged between the CE 100 and FE 110 according to an interval set during the binding phase.

During the shutdown or unbinding phase, the FE 110 or the CE 100 send an unbind message 138 to the other which ends their association. In Figure 3 the unbind message 138 is shown being sent from the FE 110 to the CE 100, although the CE 100 could have sent the unbind message 138 to the FE 110.

The information exchange between the CE and FEs can have several embodiments, which are FE or interconnect specific. The inventive protocol provides mechanisms for information exchange that provides reliability (using TCP/IP), robustness against DoS attacks (separation of control and data channel), extensibility (separation of protocol & data model) and low overhead/good performance (using TLV encoding).

Referring now to Figure 4 an NE (in this example a router) 200 which implements the present protocol is shown. The NE 200 includes three FEs 110a, 110b and 110c as well as a CE 100 and an interconnect 150. In one embodiment the interconnect 150 is realized as a switch fabric. Host 160 sends a packet to FE 110a on port-1. The packet is forwarded from FE 110a to FE 110c. FE 110c receives the packet and forwards it over

the egress port to host 170. In the NE 200, the present protocol is used to exchange information between the FEs 110a and 110c by way of CE 100.

The process starts with the CE 100 performing a bind operation with FE 110a, FE 110b and FE 110c. As a result of the bind operations a data channel is established between the FE 110a and CE 100, as is another data channel between CE 100 and FE 110c. A control channel is established between the FE 110a and the CE 100, as is another control channel between the CE 100 and the FE 110c. As described above, the capability discovery phase is executed as is the configuration operation phase. configuration request messages are used to download any configuration information from the CE 100 to the FEs 110a and 110c and the configuration response messages are used to notify the CE 100 about the result or status of the configuration requests.

A flow chart of the presently disclosed method is depicted in Figure 5. The rectangular elements are herein denoted "processing blocks" and represent computer software instructions or groups of instructions. The diamond shaped elements, are herein denoted "decision blocks," represent computer software instructions, or groups of instructions which affect the execution of the computer software instructions represented by the processing blocks.

Alternatively, the processing and decision blocks can be performed by functionally equivalent circuits such as a digital signal processor circuit or an application specific integrated circuit (ASIC). The flow diagrams do not depict the syntax of any particular programming language. Rather, the flow diagrams illustrate the functional information one of ordinary skill in the art requires to fabricate circuits or to generate computer software to perform the processing required in accordance with the present mechanism for exchanging information between a control element and a set of forwarding elements. It should be noted that many routine program elements, such as initialization of loops and variables and the use of temporary variables are not shown. It will be appreciated by those of ordinary skill in the art that unless otherwise indicated

herein, the particular sequence of processing blocks and/or decision blocks described is illustrative only and can be varied without departing from the spirit of the mechanism for exchanging information between a control element and a set of forwarding elements. Thus, unless otherwise stated the processing blocks and/or decision blocks described below are unordered meaning that, when possible, the processing blocks and/or decision blocks can be performed in any convenient or desirable order.

Referring now to Figure 5, the process starts and processing block 310 is executed. In processing block 310 a binding phase is performed between the FE and the CE. In the binding phase, the FE sends a bind request to the CE. In response to receiving the bind request, the CE sends back a bind response to the FE. This step also signifies the establishment of the control channel between the FE and the CE.

In processing block 320 a data channel is established between the CE and the FE. During the binding phase, encapsulation information is exchanged between the CE 100 and the FE 110, which enables the provision of a separate data channel for the exchange of data packets between the CE and the FE. The GRE protocol, for example, may be used for data channel formation.

In processing block 330 the capability discovery phase is executed wherein the CE sends a capability request to the FE. The FE, in response to receiving the capability request, sends back a capability response with its capability information to the CE. As part of the capability discovery phase the CE also sends a topology request to the FE. The FE responds to the topology request with a topology response which includes the FE's topology information relative to other FEs. If the CE deems the FE's capabilities and topology acceptable and if the CE is ready to control and configure the FE, the CE sends a Start FE Operation message to the FE. After the Start FE Operation message is received by the FE, the FE can report events and send packets to the CE.

Processing block 340 is performed next, wherein the configuration operation phase is performed. In this phase a configuration request is sent from the CE to the FE. In response to receiving the configuration request, the FE sends back a configuration response to the CE. The CE also sends a query request to the FE. In response to receiving the query request, the FE sends back a query response to the CE. Packet redirection between the CE and the FE also takes place over the data channel.

In processing block 350 an unbind of the CE and the FE takes place. The FE or the CE sends an unbind message to the other which ends their association. After the unbind, the process ends.

The present protocol provides a mechanism for exchanging information between the control element and one or more forwarding elements in a network architecture. Some of the features of this mechanism include extensibility (achieved by separation of the protocol and the data model), reliability (by using TCP/IP), improved performance (by using TLV encoding) and robustness against Denial of Service (DoS) attacks via separation of the Data and Control channel.

Having described preferred embodiments of the mechanism for exchanging information between a control element and a set of forwarding elements it will now become apparent to those of ordinary skill in the art that other embodiments incorporating these concepts may be used. Additionally, the software included as part of the mechanism for exchanging information between a control element and a set of forwarding elements may be embodied in a computer program product that includes a computer useable medium. For example, such a computer usable medium can include a readable memory device, such as a hard drive device, a CD-ROM, a DVD-ROM, or a computer diskette, having computer readable program code segments stored thereon. The computer readable medium can also include a communications link, either optical, wired, or wireless, having program code segments carried thereon as digital or analog signals. Accordingly, it is submitted that that the mechanism for exchanging information between

a control element and a set of forwarding elements should not be limited to the described embodiments but rather should be limited only by the spirit and scope of the appended claims.